C# Design Patterns: Rules Engine Pattern

APPLYING THE RULES ENGINE PATTERN

Steve Smith
FORCE MULTIPLIER FOR DEV TEAMS

@ardalis | ardalis.com | weeklydevtips.com

Objectives

What is the rules engine pattern?

What problems does a rules engine solve?

What is the structure of the rules engine pattern?

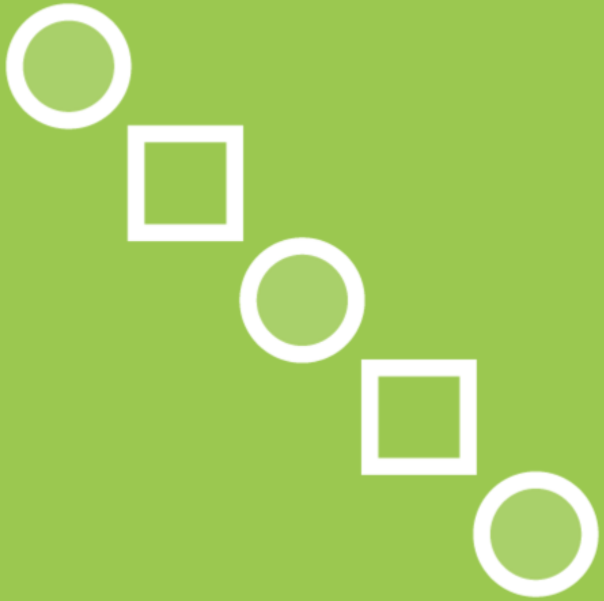
How to apply the pattern in real code?

How to recognize related patterns?

What Is the Rules Engine Pattern?

A rules engine processes a set of rules and applies them to produce a result.

A rule describes a condition and may calculate a value.



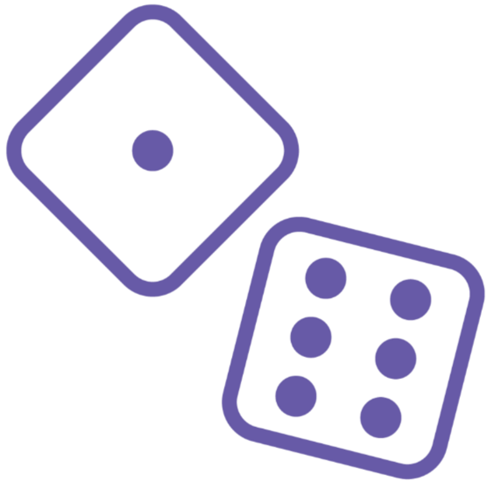
A Rules Engine is a *behavioral* design pattern.

Related Courses

Refactoring

SOLID Principles

Examples of Operations



Scoring games

Calculating Discounts for Customer Purchases

Diagnosing Health Concerns

(and other expert systems)

What Problem Does a Rules Engine Solve?

The Open/Closed Principle

Code should be open to extension, but closed for modification

Adding more complexity repeatedly may be a sign a rules engine could help

Prefer maintaining existing software through new classes

Defining Rules

Each rule you extract should follow Single Responsibility Principle

Rules are managed using an engine that chooses which rule(s) to apply

Rules may be ordered, aggregated, or filtered as appropriate

Demo

A simple discount calculator

New Promotions!

Customers get 10% off on their birthday!

Loyal customers get an extra 10% off on their birthday!

Demo

Implementing new discounts

Analysis

Method keeps growing in complexity

- Cyclomatic Complexity > 10

Any change must be made in this method

Violates Open/Closed Principle

What Is the Structure of the Rules Engine Pattern?

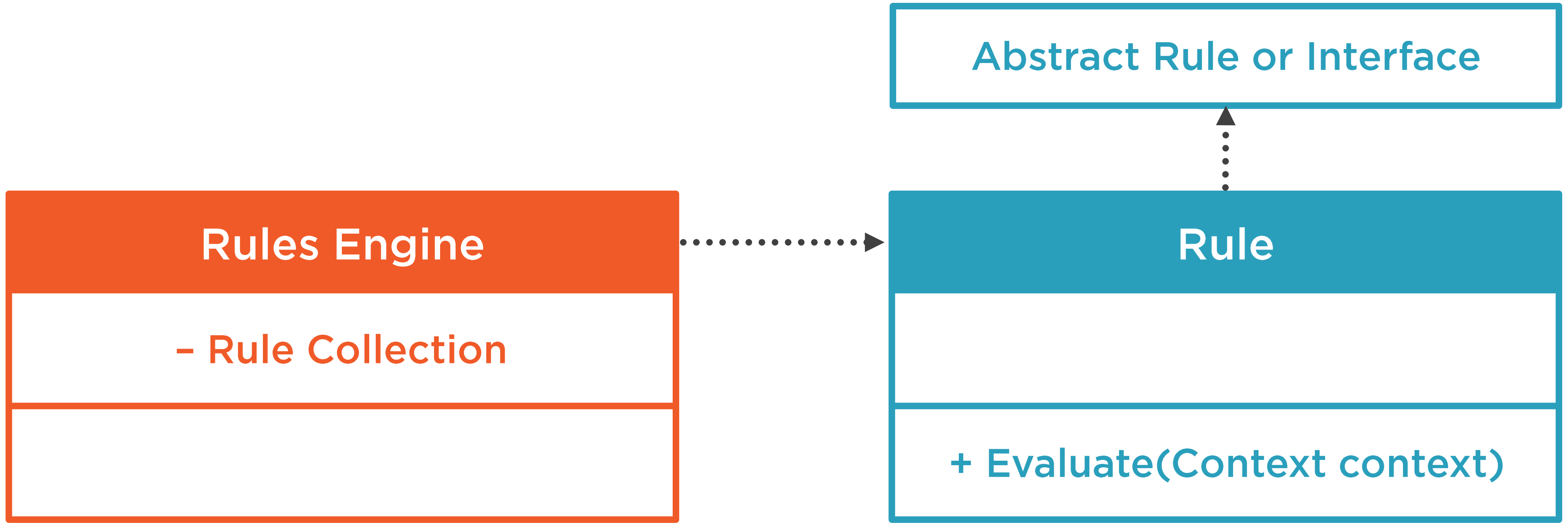
Rules Engine Collaborators

Rule Collection

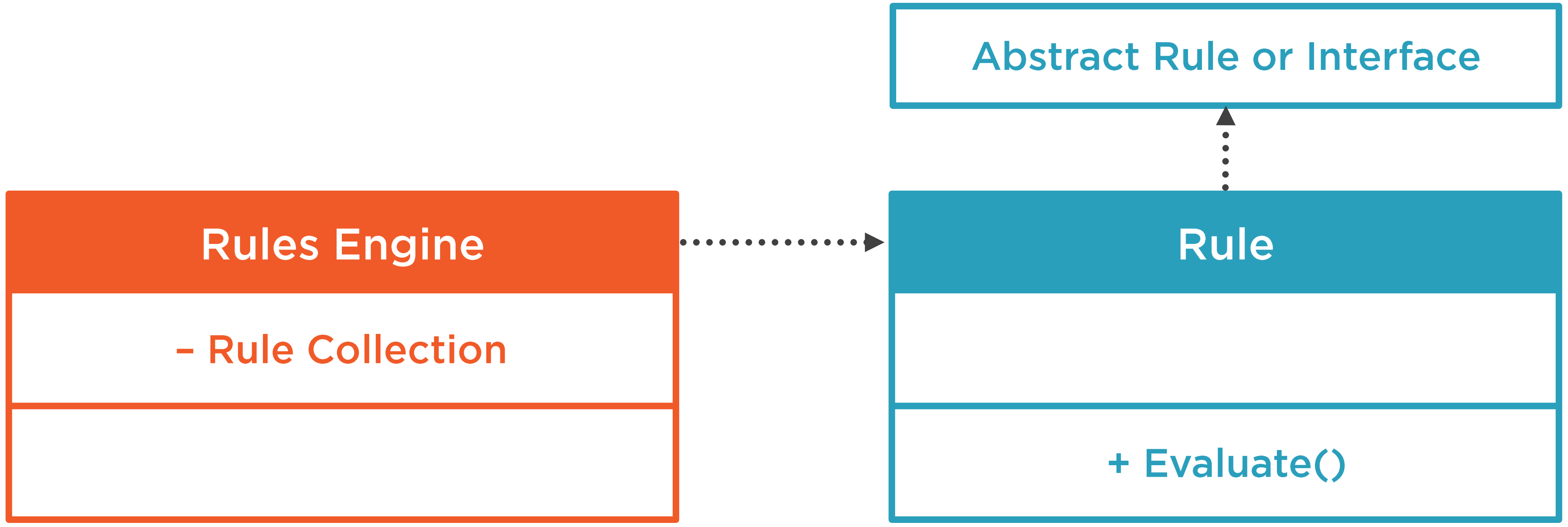
Rules Engine

System Input (or Context)

Rules Engine Structure



Rules Engine Structure



Working with Rules

Keep individual rules simple

Allow for complexity through combinations of simple rules

Decide how rules will combine or be chosen

Consider whether rule ordering will matter in evaluation

Implementing a Rules Engine

- Accept rules collection in engine constructor
- Allow adding/removing rules or swapping sets of rules via methods
- Apply the rules to a given context or system state
- Choose correct rule to apply or aggregate rules

How Do We Apply a Rules Engine to Existing Code?

Steps to Apply Rules Engine

Follow refactoring fundamentals

Extract methods for individual conditions

Convert methods into Rule classes

Create Rule Engine and evaluate Rules

Replace original method logic with call to Rules Engine

Demo

Applying the Rules Engine pattern to Customer Discounts

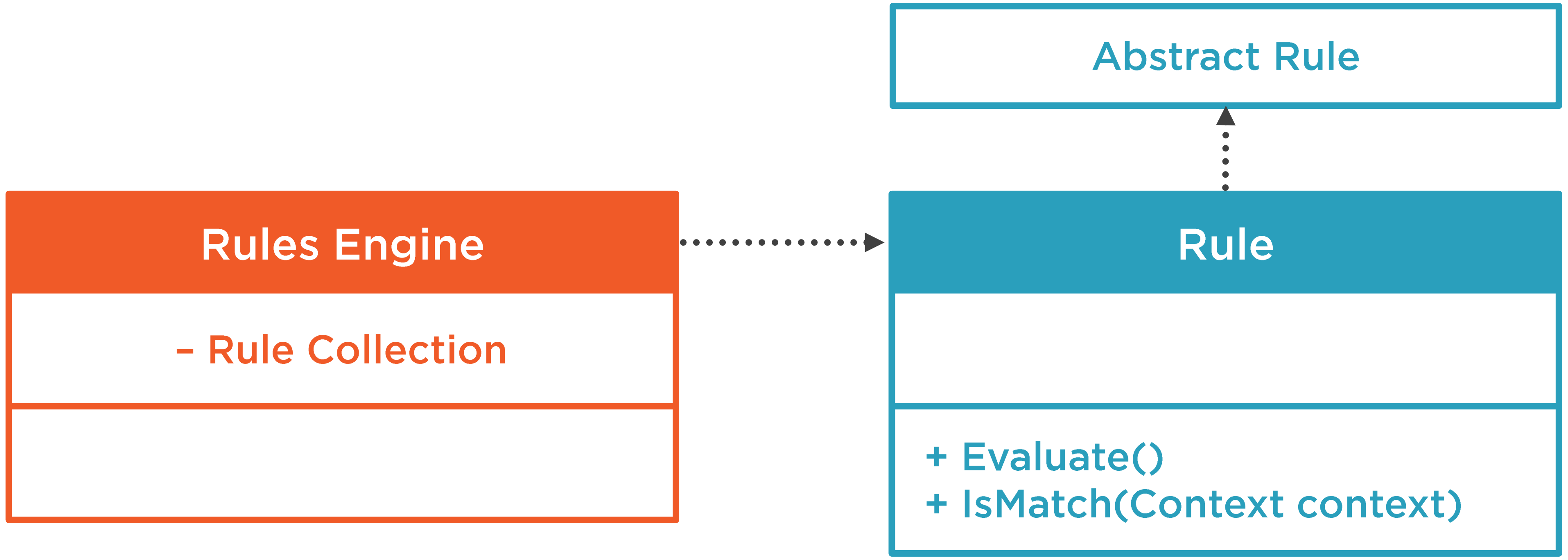
Analysis

Each rule class is very simple

Complexity of calculating discounts is now very small

Adding new discount scenarios should only require adding new classes (Rules)

Alternate Rules Engine Structure



Considerations for Rules

Typically Read-Only

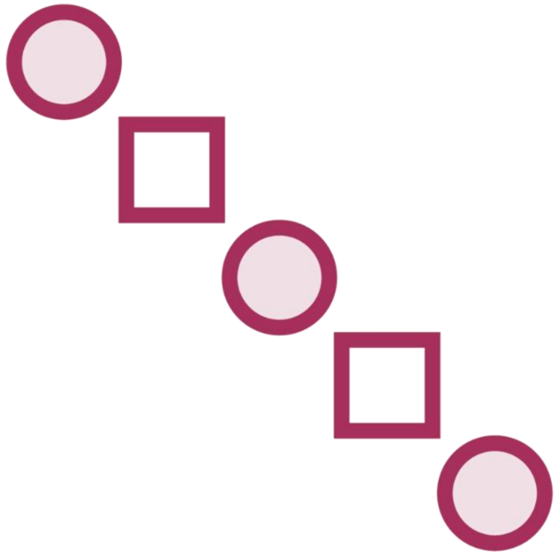
Dependencies between Rules?

Explicit Sequence or Priority

Short-circuiting execution

Managing Rules with Persistence and UI

Related Patterns



Specification Pattern

- A Domain-Driven Design Pattern
- Describes a query in an object

Key Takeaways

Rules Engine Design Pattern

- Behavioral Pattern
- Split up conditional logic into explicit rule classes

Common uses:

- Calculating Scores
- Calculating Discounts
- Business Logic Calculation

Key Takeaways

Structure

- Engine
- Individual Rules
- Strategy to Select or Combine Rules

Consider:

- Read-only rules
- Rule dependencies
- Rule priority and short-circuiting
- Rule management

C# Design Patterns: Rules Engine Pattern

APPLYING THE RULES ENGINE PATTERN

Steve Smith
FORCE MULTIPLIER FOR DEV TEAMS
@ardalis | ardalis.com | weeklydevtips.com